Specification and Assessment of Temporal Requirements using Simulink Test

Jean-Francois Kempf, Khoo Yit Phang and Akshay Rajhans
MathWorks
1 Lakeside Campus Drive, Natick, MA 01760.
{jkempf, ykhoo, arajhans}@mathworks.com

I. Introduction

Cyber-physical systems (CPS) are often developed using Model-Based Design with computational models used throughout the design process. Over the past decade, specification formalisms such as various temporal logics have emerged as mechanisms for rigorous and unambiguous monitoring and testing of models. Yet, writing temporal logic formulas tends to be unintuitive and difficult for engineers and practitioners. In this abstract we present a new functionality in Simulink® Test™ for authoring and assessing temporal specifications integrated in the MATLAB® and Simulink® environment without the need for explicitly writing out logical formulas.

II. Overall Architecture

Simulink Test[1] provides tools for systematic simulation-based testing of models, generated code, and simulated or physical hardware. Simulink Test Manager[2] helps the users manage, author, and execute tests.

Starting with Release 2019a, Simulink Test offers a functionality for authoring logical and temporal specifications. A Simulink model is specified as the system under test (SUT). Temporal specifications can be authored using structured English from pre-existing templates and patterns. Assessment of specifications invokes a simulation of the SUT model. Pass/Fail results are shown with graphical depictions of expected and observed results, textual explanations, and a hierarchical subexpression assessment tree.

III. Authoring Temporal Specifications

Figure 1 shows a window for authoring logical and temporal specifications with three key components.

A. Use of Structured English to Author Logical Specifications

Clicking on 'Add Assessment' lets the user choose from pre-existing templates and patterns to construct a specification. The top specification shows a *bounds check* pattern in which the specified signal `signal` is expected to always stay within the bounds `lowerBound` and `upperBound`.

The bottom specification is of the *trigger response* pattern where if the trigger (`driverInput > driverInputAmplitude*stepRatio`) ever holds, then within the specified period after that (`with no delay`), the specified response (`abs(signal - signalRef) < overshootTolerance) must stay true for tau seconds`) holds.

The bottom specification is shown in its collapsed form where it reads like an English language specification. Yet, because it is a logical formula under the hood that is constructed from a tree-like structure (shown uncollapsed for the top specification), it is rigorous, formal, and unambiguous by construction. Hyperlinks `Speed Tolerance` and `Prevent Overshoot` point to corresponding informal natural language requirements authored using Simulink® Requirements™.[3]

B. Visual Representation

The top right corner of Figure 1 shows a visual representation along with a fictitious trace that would pass the specification.

C. Symbol Resolution

The bottom right corner of Figure 1 shows the symbol table where the named symbols used in specifications, such as `signal`, `driverInputAmplitude`, and `overshootTolerance`, are mapped to a Simulink signal, a MATLAB workspace variable, and a MATLAB expression respectively.

IV. Assessment of Temporal Specifications

Figure 2 shows the assessment result window for a specified assessment along with its symbol table shown at the top of the main pane. Underneath that are **Expected Behavior**: a graphical representation of expected behavior given a formula, **Actual Result**: a graphical depiction of the observed simulation result, and a textual **Explanation** about why at certain points in time the assessment failed. The **Expression Tree** shows hierarchical subexpression assessment results. Simulation Data Inspector[4] enables the inspection of data values at various points in time within each assessment plot.

Acknowledgments

We acknowledge helpful discussions with Krishna Balasubramanian, Paul Urban, and Pieter Mosterman.

[1] https://www.mathworks.com/products/simulink-test.html
[2] https://www.mathworks.com/help/sltest/ug/introduction-to-the-test-manager.html
[3] https://www.mathworks.com/products/simulink-requirements.html
[4] https://www.mathworks.com/help/simulink/slref/simulationdatainspector.html

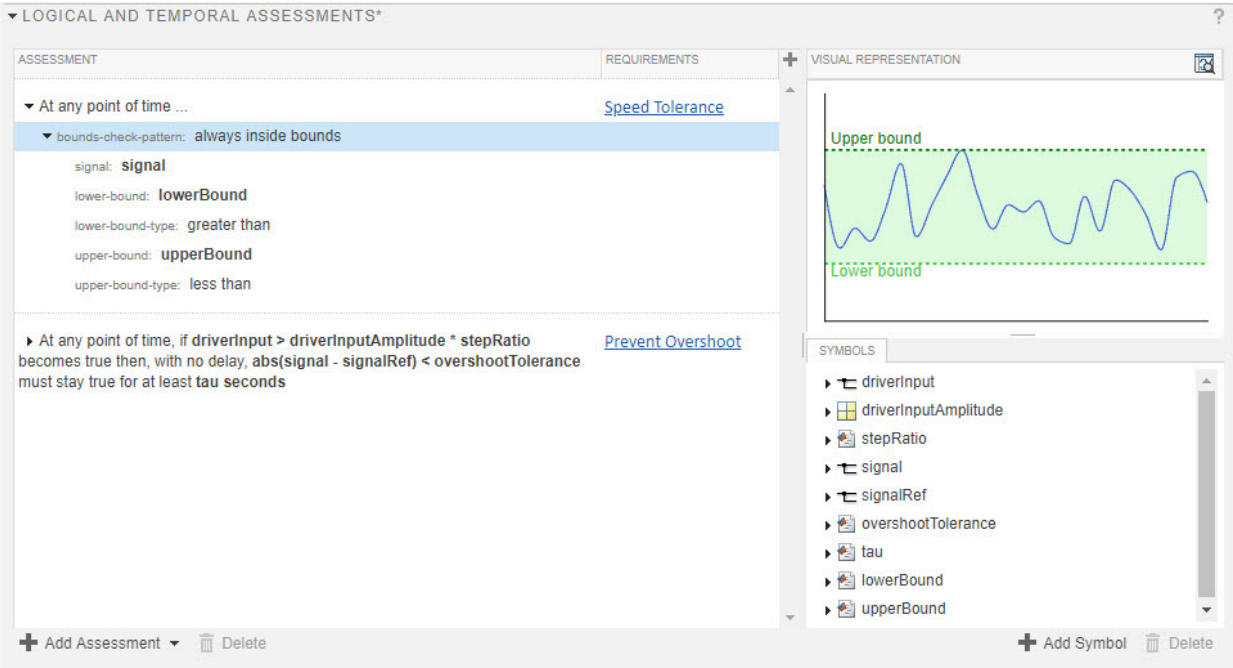

Fig. 1. Authoring Temporal Specifications.

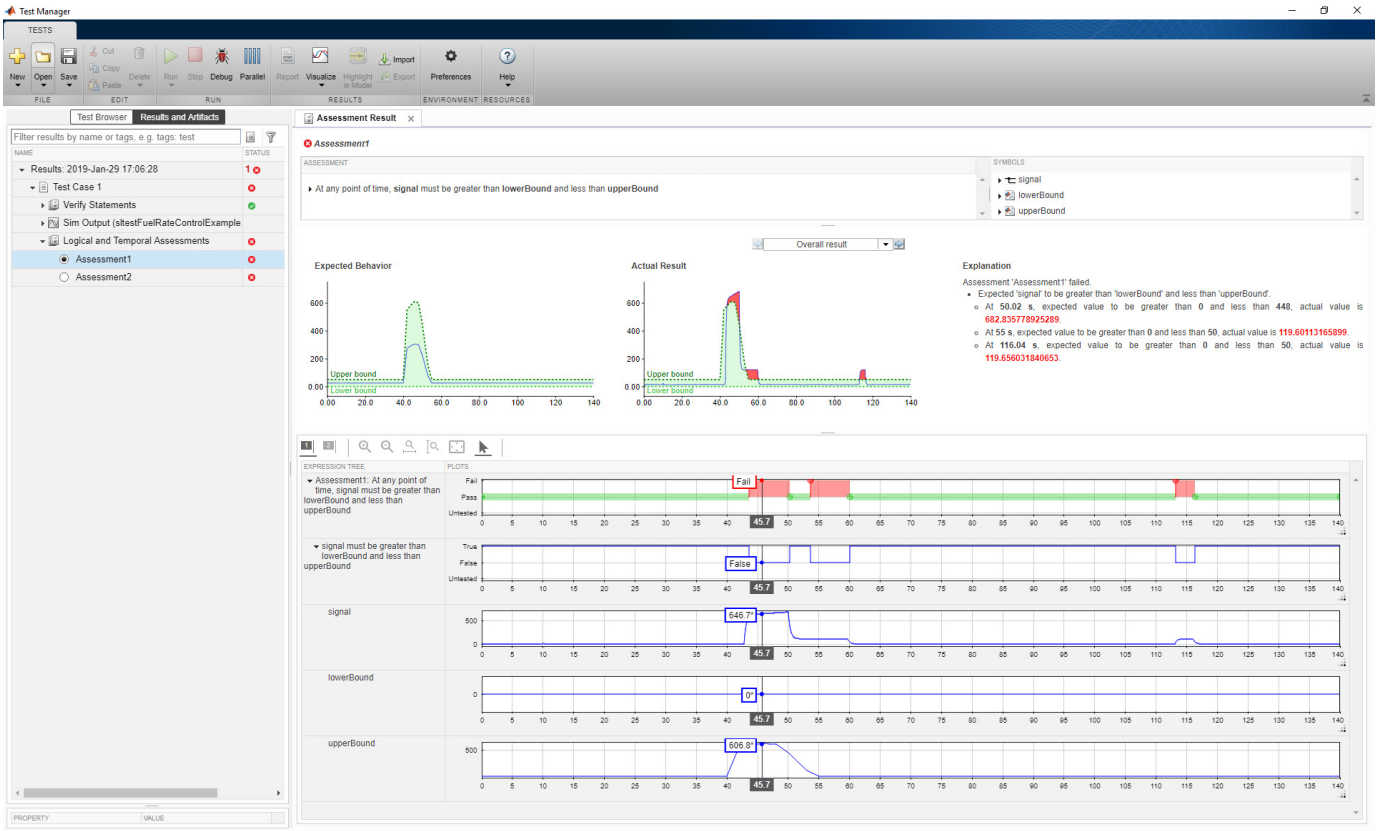

Fig. 2. Assessment of Temporal Specifications.